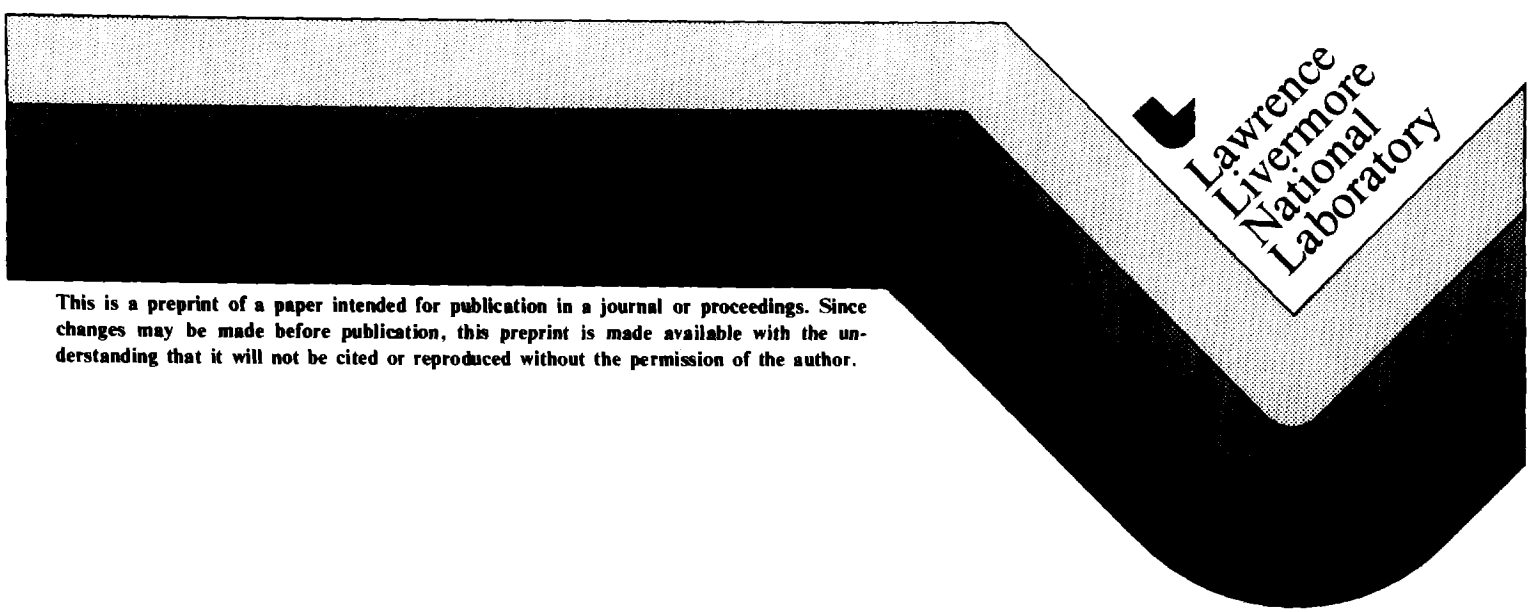
UCRL- 94415
PREPRINT

STRIVING FOR "NATURAL" USAGE

This paper was prepared for presentation at
ACM/SIGCHI '86
Human Factors in Computing Systems
Boston, Massachusetts
April 16, 1986



Striving for "Natural" Usage†

Nancy A. Storch
Applications Systems Division
Computations Department
Lawrence Livermore National Laboratory

P. O. Box 808
Livermore, Ca. 94550
(415) 422 - 8942

Abstract

During the past year, the Non-Destructive Evaluation Section has supported developing a user environment for their technicians and scientists who do image processing and analysis of images obtained from various sources. In particular, this has involved bringing together certain software tools which already existed into a prototype, trying to understand user needs by observation, discussion, insightful guessing, and then continuing on to build something better.

One of our primary goals has been to achieve a *natural* user interface. We have found this to be most difficult to accomplish. A natural interface implies a certain simplicity, while still offering all that is necessary to the user's task. Simple on the outside hasn't meant simple on the inside. Our efforts have required internal complexity in handling: pointing of the user to the image or screen and resultant actions, device independent graphics and raster operations, application menus, working image data base and command recognition. These are, for the most part, independent of the image processing functions.

Starting from broad, not very well defined requirements, it has become an on-going effort to checkout both interactive and image processing operations. A framework was developed to allow for flexibility in image processing operations and their integration into work-related applications. The results of our work indicate that success in accomplishing an easy, natural usage is due to: commitment and concentration on usage, continual reevaluation during development, and allowance for the overhead of "tidying up" to maintain internal order and clarity.

March 25, 1986

† Work performed under the auspices of the U. S. Department of Energy by the Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48.

Striving for "Natural" Usage

Nancy A. Storch
Applications Systems Division
Computations Department
Lawrence Livermore National Laboratory

P. O. Box 808
Livermore, Ca. 94550
(415) 422 - 8942

1. Introduction

This paper describes an approach used in developing a user environment for image processing and analysis and the program which is used. The intended users are approximately a dozen technicians and scientists who work as part of a support team which inspects parts for flaws and imperfections. The images with which they work have been obtained from either ultrasonic scanning or radiography. An ultrasonic image is not related to the spacial geometry of the object scanned. It depicts echo responses. Ultrasonic imaging is especially useful in detecting cracks or misalignments. A radiographic image, on the other hand, is spacially related to the object. The greyscale shadows measure the intensity of the radiation after passing through the object. Radiographs are useful in detecting bubbles or non-uniformities in material density. Although ultrasonic and radiographic scanning of the same object result in very different images, the same image processing tools can be applied to both types of images. It is the expert who must decide what and how scannings are to be done, and what image processing is required in the examination and evaluation of the results.

The technicians and scientists are experts at what they do and typically have many years of experience. Most of these people are not computer-wise, but are forced to use the computer as one aspect of their job. Their image processing needs are varied and sporadic. Months may go by between a particular user's sessions and when necessary he/she may need to work with the program intensely for a period of a week or more. Because the user requirements are broad and not very well defined, we decided to create an appropriate user environment which could offer suitable image processing tools. This environment had to be somewhat adaptable to each user and the job to be performed. Continuing programming support was anticipated for the improvement of the user-interface and for inclusion of additional tools as they would be needed. Our commitment to satisfy these users' needs caused us to focus our attention on usage, resulting in an emphsis more on direction than on the specifics of the solution.

2. "Natural" Usage and Our User Interface

We are striving toward the ideal of natural usage, which may be considered equivalent to a "perfect 10" when describing the amount of ease the intended users feel when working with the program. The program having natural usage contains the appropriate tools to accomplish the desired task; and the user can get at the tools in a way which is natural. Not only does the user find the things which he/she needs to do the job easily at hand, but the program assists the user in ways which are very compatable with that person's physical and mental capabilities. The steps and actions the user goes through in accomplishing his/her work need to be simple, direct and easy. Unnecessary steps and actions which may arise from implementation, hardware or system pecularities need to be minimized. The user wants to succcessfully complete command scenerios

without needing to refer to overly detailed instructions. The program has to be responsive so that the user is kept aware of what is going on at all times in order to maintain proper synchronization between the user and the program. The actualization of natural usage may have some negative effects if the closeness between user and program becomes too extreme; this seems far fetched at this time.

In striving for natural usage, we turned to developing a natural user interface. The user interface goals deal with three areas of communication between the human being and the computer. The idea is to extend and complement certain human capabilities with suitable matching program capabilities. First, we are striving for an extension in the use of the hand. With tools such as mouses and strategically placed function keys we have eliminated most of the need for typing when pointing at menus and manipulating images, thereby reducing the amount of work required to perform simple image processing functions. We have not been able to eliminate typing altogether since it is necessary to supply alphanumeric information such as filenames. On the other hand, for those who are skilled typists and prefer entering commands directly, we allow menus to be bypassed and commands either typed in or read from a command file. Although commands have a short form for quick typing, a long form may be preferred in the construction of command files for better documentation.

Second, we are striving for an extension of the mind's eye. We want to assist the user in visualizing the images which are being worked with. Examples are the ways in which: images are displayed, a subset of an image is defined, coloring is applied and changed, and image data is selected for viewing . And third, we are striving for an extension of the mind's analytic ability, where the user can guide the computer in transforming images through image data calculations. This together with the viewing of the results, enables the user to analyze materials for consistency and flaws. We are working on providing the right transformation tools and letting users know that they are available and how to use them.

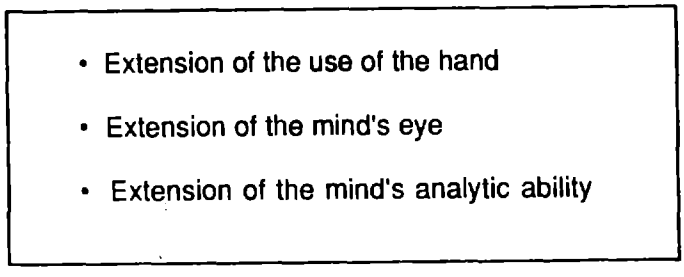


Figure 1: User interface goals.

Our user interface also has to accomodate the transition that occurs in the more frequent user whose learning curve takes him/her from simple operations to more complex operations. This user welcomes the ability to bypass the menu and directly use the command language. The even more sophisticated user can group a series of commands into a file and execute them as a single unit. For those who rely on the menu, we use a minimum of layering and make it easy to enter and exit menus. Menu selection is accomplished by using a pointer and the return key or by striking an equivalent special key. When in menu mode, operations are easily repeated with repeated striking of the return key.

3. Development Cycle

The development cycle reflected our desires to understand user needs by observation and discussion. A fair amount of insightful guessing was added as the new program was built. Figure 2 gives an overview to the development cycle.

First certain existing software tools were brought together into a prototype. The usage of the prototype was studied, and also the usage of other software available to users. This lead to making judgements as to what functions to include in the new program and how they would be used. It also indicated that some changes in hardware were needed. Questions raised and answered again and again were: "What is important?", "What is necessary?", "When is it necessary?", "What is the simplest way to do this?" The results were that some user interactions were dropped, some kept. For example, it was found to be better to have the menu displayed on the terminal, rather than on the screen used for the image; however moving around the menu with up and down pointers was accepted and a second set of pointer keys was added for compatibility with a familiar program. It was observed that users were comfortable with the way an object is moved and scaled in Macintosh drawing programs. Therefore, similar capability was supplied where the program responds based on the position of the cursor relative to the boundary of the object.

In order to be more sensitive to user feedback, certain freedoms in development were allowed that are not standard. These gave us the flexibility to apply what we learned as we developed the program. For example, we did a general analysis and specification, then moved on to complete relatively small pieces at a time and check out the results of the user interactions as soon as possible. Detailed documentation was delayed until positive user feedback was received and any changes had been made.

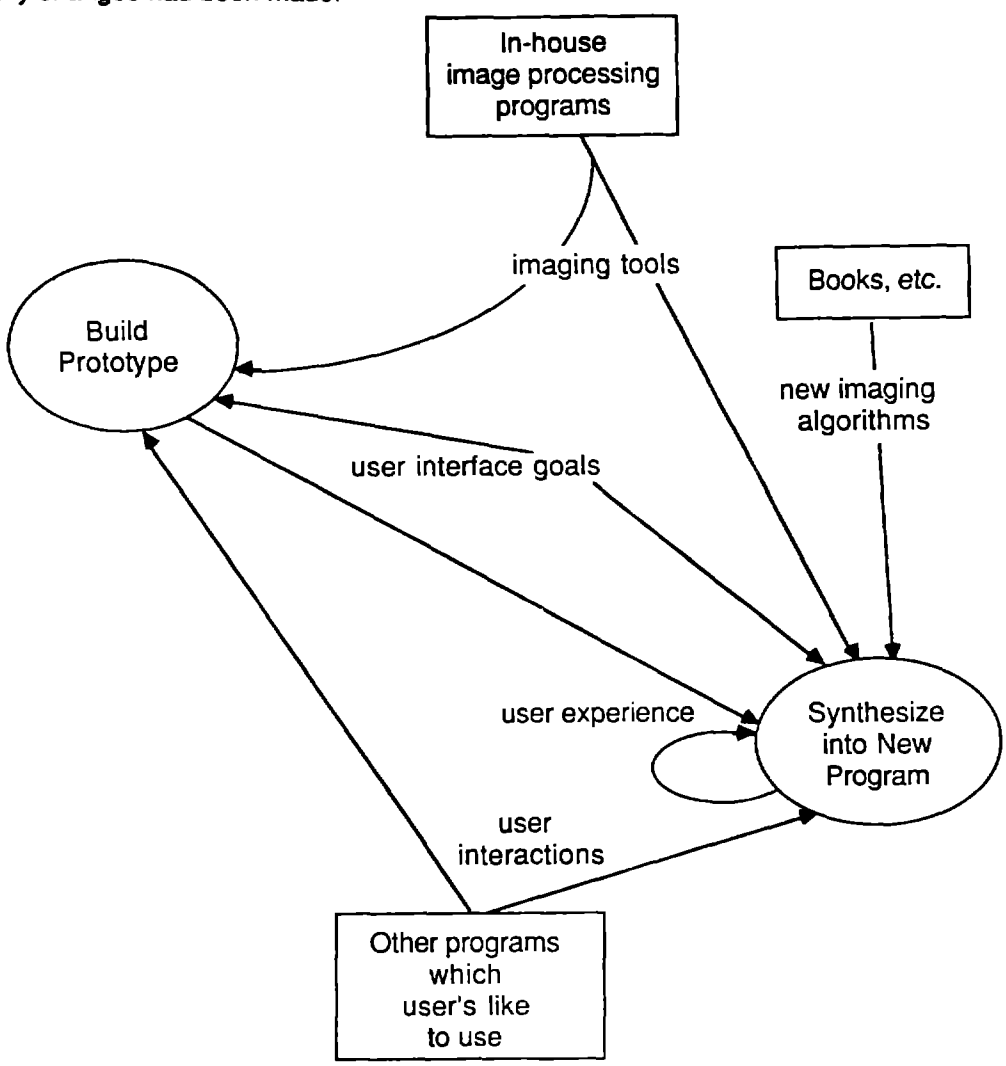


Figure 2: Development of the program.

The screen surface was conceptualized as being composed of images and related overlayed information. A skeleton program was written which recognized some general shell commands, image transfer commands, and graphics commands. These required the low level utitlies which do: image management, raster graphics, and command parsing . Then came image processing commands and interactive applications which required utilities to do: image processing transformations, image interaction and menu handling. Applications were built from existing commands and sometimes resulted in new commands being added. We are currently interfacing a new workstation with higher resolution graphics screen and replacing the terminal keypad cursor with a mouse. Referring back to the initial plans, it was found that some areas needed to be expanded immediately while others could be delayed.

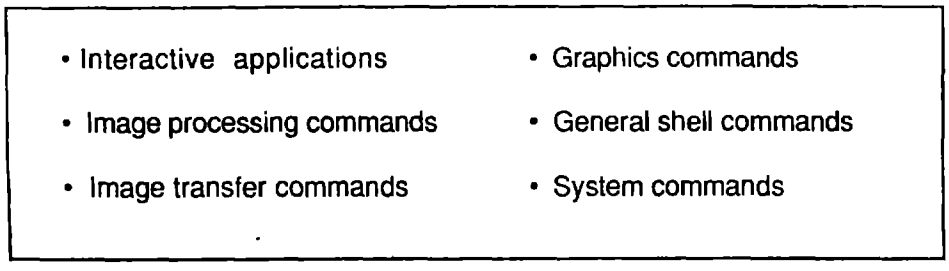


Figure 3: Classes of commands

The flexibility and emphasis on user feedback lead to simultaneous top-down and bottom-up development of certain capabilities with unavoidable postponement of others. The focus has been to follow through and complete one application at a time. In the process of understanding and implementing an application, supporting commands become more clearly understood also. So it can be expected that commands may be modified, added, deleted, combined with others based on a better understanding of what is needed. One needs to allow for still more changes which may become apparent with changing users needs. It is important to reflect the changes in all current documentation and to clean up software to keep it well designed for future maintenance, as the quantity of software needs to be kept manageable. Thorough checkouts of both interactive and image processing operations help to keep the quality and reliability up. During testing we substitute easy recovery for high reliability.

4. Program Structure

The program structure had to be simple to allow for flexibility in image processing operations and their integration into work-related applications. Commands and applications are independent of each other so that they can be easily added or changed. Utilities have several levels. At the highest level they are independent of hardware and system. Also the help library is independent of the program and is easily updated. Figure 4 shows the basic program structure.

Simple on the outside hasn't meant simple on the inside. Our efforts have required internal complexity in handling: pointing of the user to the image or screen and resultant actions, device independent graphics and raster operations, application menus, working image data base and command recognition. These are, for the most part, independent of the image processing functions.

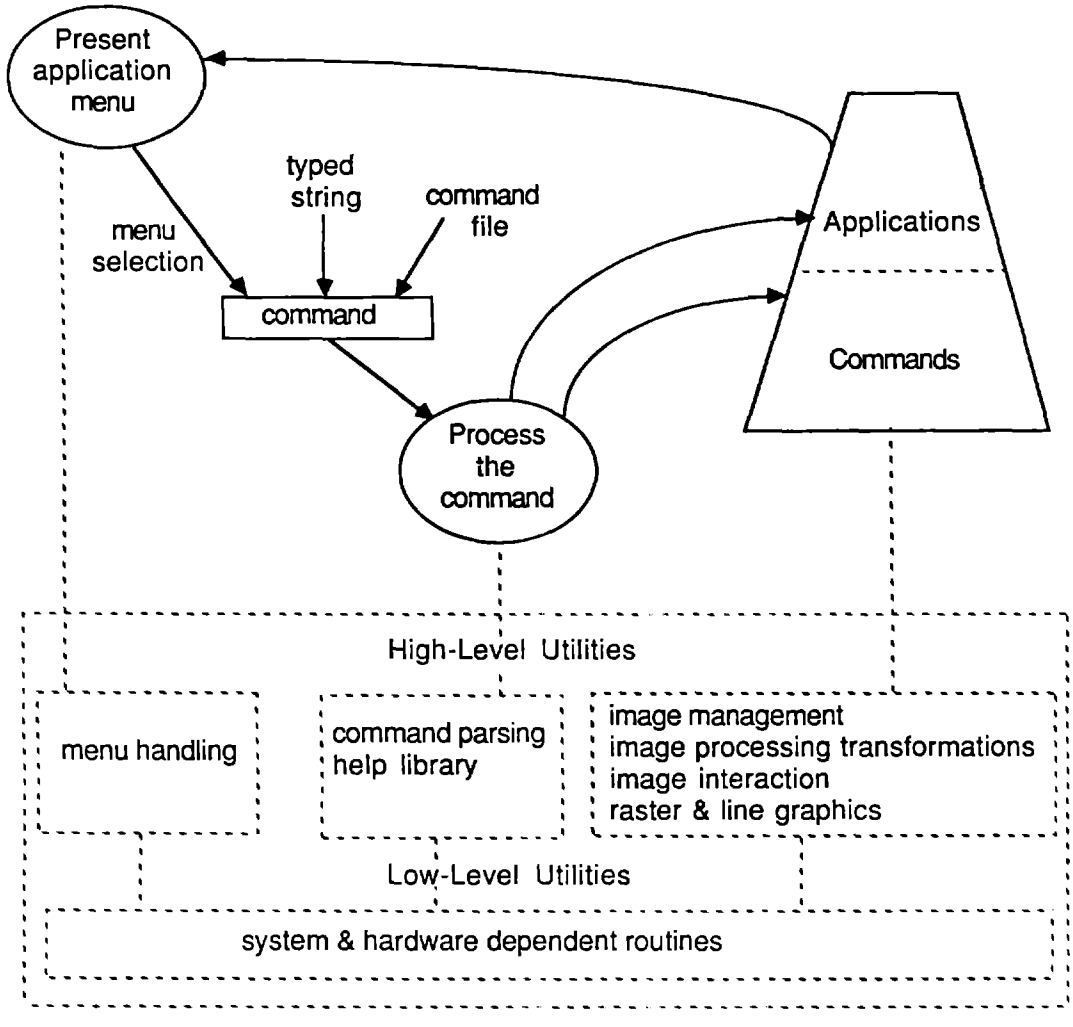


Figure 4: Program structure.

5. One Application

The pseudocoloring application is used to do contrast enhancement. It presents to the user in a menu form certain combinations of commands. With this application a user can calculate and overlay a histogram of a subimage, overlay a user defined colorbar, and/or change pixel color assignments. The histogram and colorbar allow the user to subjectively improve the contrast of the portion of the image to be enhanced by serving as color reference guides against which to measure. A sequence of operations are selected by the expert user. The menu lists the options in logical groupings, with the most likely toward the top. Exactly how the final result is arrived at is not known.

The success of this application depends on correctly identifying what the users need to see and manipulate in order to easily improve image contrast with enhancement tools. User experience with a prototype indicated a priority of actions (later reflected in the ordering of selections presented in the menu) and gave positive feedback on how to best present the histogram (colors and labeling) in conjuction with the image.

6. Conclusions

The results of our work indicate that success in accomplishing an, easy, natural usage is due to: commitment and concentration on usage, continual reevaluation during development, and allowance for the overhead of "tidying up" to maintain internal order and clarity. Our loose methodology came from our emphasis on learning through users experience over standard software development. Constraints which influenced our efforts were: the terminal and graphics hardware available, the operating system, the need to provide something to users in a reasonable time frame, and only two people to do the work. Although we have made compromises at times, our results have been positive. We believe that we are moving in a direction toward natural usage.